



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
-----------------	-------------	----------------------	---------------------	------------------

10/620,929

07/16/2003

Lawrence B. St. Clair

200206325-1

8857

22879

7590

07/17/2006

HEWLETT PACKARD COMPANY  
P O BOX 272400, 3404 E. HARMONY ROAD  
INTELLECTUAL PROPERTY ADMINISTRATION  
FORT COLLINS, CO 80527-2400

EXAMINER

CHAUDRY, MUJTABA M

ART UNIT

PAPER NUMBER

2133

DATE MAILED: 07/17/2006

Please find below and/or attached an Office communication concerning this application or proceeding.



## **DETAILED ACTION**

Claims 1-10, 15-19, 21 and 22 are currently pending. Applicant is suggested to cancel non-elected claims in subsequent communication.

### ***Information Disclosure Statement***

The references listed in the information disclosure statement (IDS) submitted on July 16, 2003 have been considered. The submission is in compliance with the provisions of 37 CFR 1.97.

### ***Oath/Declaration***

The Oath filed July 16, 2003 complies with all the requirements set forth in MPEP 602 and therefore is accepted.

### ***Drawings***

The drawings submitted July 16, 2003 are accepted.

### ***Specification***

The specification submitted July 16, 2003 is accepted.

### ***Allowable Subject Matter***

Claims 3, 5-10, 17 and 22 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the

Art Unit: 2133

base claim and any intervening claims and provided that the rejections under 35 USC 112 for the base claims have been resolved.

### ***Claim Objections***

Claim 15 is objected to because of the following informalities:

- In the preamble it is not clear what the system/arrangement is for.

Appropriate correction is required.

### ***Claim Rejections - 35 USC § 112***

The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

Claim 1 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

- It is not clear from the claim (lines 9-12) if the “plurality of platform dependent modules” are adapted to receive the data in serial or in parallel.
- The title of the application refers to “platform independent” while the specification and claims refer to “platform dependent”. It is not clear if Applicant has inadvertently interchanged the two. Clarification is requested.
- Is the “error detection selection module” same as the “error detection module”? Also, in line 15 what is the “at least one” referring to?

Art Unit: 2133

- Generally, lines 15-32 are not clear in what the Applicant trying to convey. For example, "...to effectuate at least the selected error detection scheme..." and what is the "...at least one feature..." in line 21.

Claim 2 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

- It is not clear from the claim how step (i) is any different from step (ii), since "file executed by the target platform" and "information accessed by a program that runs on the target platform" are essentially the same procedure.

Claim 4 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

- It is not clear what is meant by "at least momentarily" since momentarily is a relative concept.

Claim 5 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

- In line 2, "the at least one feature" is not clearly defined.
- In line 3, the limitation "adapted to provide pertains to one or more of" is vague and indefinite.

Art Unit: 2133

Claim 15 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

- In line 11, it is not clear what the “at least one feature” is referring to, as it is not defined.

Claim 18 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

- In line 9, it is not clear what the “at least one feature” is referring to, as it is not defined.

Claim 21 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

- In line 28, it is not clear what the “at least one feature” is referring to, as it is not defined.

For the purposes of rejection, the claims will be interpreted in accordance with MPEP 2111 and with certain assumptions, which may or may not be what Applicant initially intended.

***Claim Rejections - 35 USC § 103***

The factual inquiries set forth in *Graham v. John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35

U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

Claims 1, 2, 4, 15, 16, 18, 19 and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Liu et al. (USPN 6804816) further in view of Lubart et al. (USPN 552875).

As per claim 1, Liu et al. (herein after: Liu) substantially teaches (abstract) a computer platform that includes one or more processors and an operating system. The template includes one or more sequences of statements of a platform-dependent scripting language. The statements are processed by a Web server executing on the platform. The statements cause the Web server to define a plurality of standard management methods and call at least a first standard management method. The statements also cause the Web server to define page level variables for holding in memory values used as parameters for the standard management methods. The template also includes a place to insert application-specific statements for presenting network information to a Web-based client process and for receiving input from the Web-based client process. A developer inserts the application-specific statements. The resulting Web resource is stored on a Windows platform connected to the network for access by a Web server on the platform in response to a request from the Web-based client process. Using this template, robust network management applications can be developed efficiently for certain platforms, such as

Art Unit: 2133

windows platforms, with most of the application-specific development confined to generating a user interface, such as with HTML forms, to present the network information to the user and to receive user input and commands. Liu teaches (Figure 2D) in step 264, the standard network management methods are used to perform server-side verification of the data. If the data cannot pass verification, then control passes to step 266 which generates and displays an error message. If the data do pass server-side verification, then control passes to step 268 to continue processing.

Liu does not explicitly teach to use a specific error detection scheme to detect errors as stated in the present application.

However, Lubart et al. (herein after: Lubart) teaches, in an analogous art, (col. 2) a recovery environment by creating the recovery environment when the operating system routines are bound together to form the kernel and at operating system initialization time. This static recovery environment exists in the operating system and provides a cross reference from the instruction address of any system failure to the appropriate recovery routine. The software or hardware error detection mechanism accesses the static cross-reference to determine *which recovery routine to initiate*. Control is then passed to the appropriate recovery routine, which has access to all registers and variables in the current context of the operating system. The system allows a nested recovery function such that each of the nested functions has an opportunity to perform any recovery action required. Each recovery routine itself may have a recovery routine. Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to utilize multiple error detection schemes within the error detection process of Liu. This modification would have been obvious to one of ordinary skill in the art because one of



ordinary skill in the art would have recognized that by utilizing multiple error detection schemes would allow for specific types of errors to be detected.

As per claim 2, Liu substantially teaches, in view of above rejections, the template includes one or more sequences of statements of a platform-dependent scripting language. The statements are processed by a Web server executing on the platform.

As per claim 4, Liu substantially teaches, in view of above rejections, a computer system for developing a platform-dependent network management Web resource, the system comprising: a platform including one or more processors executing an operating system and a Web server; a computer readable memory connected to the one or more processors carrying, platform-dependent binary modules for a collection of network-device centric methods, and a platform-dependent standard binary interface for generating instances of the collection of network device-centric methods; and a template comprising one or more sequences of statements of a platform-dependent scripting language, which statements cause the Web server to perform the steps of, defining a plurality of standard management methods, calling at least a first standard management method of the plurality of standard management methods, and defining page level variables for holding in memory values used as parameters for the standard management methods; and a place to insert application-specific statements for presenting network information to a Web-based client process and for receiving input from the Web-based client process.

As per claim 15, Liu substantially teaches (abstract) a computer platform that includes one or more processors and an operating system. The template includes one or more sequences of statements of a platform-dependent scripting language. The statements are processed by a

Web server executing on the platform. The statements cause the Web server to define a plurality of standard management methods and call at least a first standard management method. The statements also cause the Web server to define page level variables for holding in memory values used as parameters for the standard management methods. The template also includes a place to insert application-specific statements for presenting network information to a Web-based client process and for receiving input from the Web-based client process. A developer inserts the application-specific statements. The resulting Web resource is stored on a Windows platform connected to the network for access by a Web server on the platform in response to a request from the Web-based client process. Using this template, robust network management applications can be developed efficiently for certain platforms, such as windows platforms, with most of the application-specific development confined to generating a user interface, such as with HTML forms, to present the network information to the user and to receive user input and commands. Liu teaches (Figure 2D) in step 264, the standard network management methods are used to perform server-side verification of the data. If the data cannot pass verification, then control passes to step 266 which generates and displays an error message. If the data do pass server-side verification, then control passes to step 268 to continue processing.

Liu does not explicitly teach to use a specific error detection scheme to detect errors as stated in the present application.

However, Lubart teaches, in an analogous art, (col. 2) a recovery environment by creating the recovery environment when the operating system routines are bound together to form the kernel and at operating system initialization time. This static recovery environment exists in the operating system and provides a cross reference from the instruction address of any system

Art Unit: 2133

failure to the appropriate recovery routine. The software or hardware error detection mechanism accesses the static cross-reference to determine *which recovery routine to initiate*. Control is then passed to the appropriate recovery routine, which has access to all registers and variables in the current context of the operating system. The system allows a nested recovery function such that each of the nested functions has an opportunity to perform any recovery action required. Each recovery routine itself may have a recovery routine. Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to utilize multiple error detection schemes within the error detection process of Liu. This modification would have been obvious to one of ordinary skill in the art because one of ordinary skill in the art would have recognized that by utilizing multiple error detection schemes would allow for specific types of errors to be detected.

As per claim 16, Liu substantially teaches, in view of above rejections, a computer platform that includes one or more processors and an operating system. The template includes one or more sequences of statements of a platform-dependent scripting language. The statements are processed by a Web server executing on the platform. The statements cause the Web server to define a plurality of standard management methods and call at least a first standard management method.

As per claims 18 and 19, Liu substantially teaches (abstract) a computer platform that includes one or more processors and an operating system. The template includes one or more sequences of statements of a platform-dependent scripting language. The statements are processed by a Web server executing on the platform. The statements cause the Web server to define a plurality of standard management methods and call at least a first standard management

method. The statements also cause the Web server to define page level variables for holding in memory values used as parameters for the standard management methods. The template also includes a place to insert application-specific statements for presenting network information to a Web-based client process and for receiving input from the Web-based client process. A developer inserts the application-specific statements. The resulting Web resource is stored on a Windows platform connected to the network for access by a Web server on the platform in response to a request from the Web-based client process. Using this template, robust network management applications can be developed efficiently for certain platforms, such as windows platforms, with most of the application-specific development confined to generating a user interface, such as with HTML forms, to present the network information to the user and to receive user input and commands. Liu teaches (Figure 2D) in step 264, the standard network management methods are used to perform server-side verification of the data. If the data cannot pass verification, then control passes to step 266 which generates and displays an error message. If the data do pass server-side verification, then control passes to step 268 to continue processing.

Liu does not explicitly teach to use a specific error detection scheme to detect errors as stated in the present application.

However, Lubart teaches, in an analogous art, (col. 2) a recovery environment by creating the recovery environment when the operating system routines are bound together to form the kernel and at operating system initialization time. This static recovery environment exists in the operating system and provides a cross reference from the instruction address of any system failure to the appropriate recovery routine. The software or hardware error detection mechanism

accesses the static cross-reference to determine *which recovery routine to initiate*. Control is then passed to the appropriate recovery routine, which has access to all registers and variables in the current context of the operating system. The system allows a nested recovery function such that each of the nested functions has an opportunity to perform any recovery action required. Each recovery routine itself may have a recovery routine. Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to utilize multiple error detection schemes within the error detection process of Liu. This modification would have been obvious to one of ordinary skill in the art because one of ordinary skill in the art would have recognized that by utilizing multiple error detection schemes would allow for specific types of errors to be detected.

As per claim 21, Liu substantially teaches (abstract) a computer platform that includes one or more processors and an operating system. The template includes one or more sequences of statements of a platform-dependent scripting language. The statements are processed by a Web server executing on the platform. The statements cause the Web server to define a plurality of standard management methods and call at least a first standard management method. The statements also cause the Web server to define page level variables for holding in memory values used as parameters for the standard management methods. The template also includes a place to insert application-specific statements for presenting network information to a Web-based client process and for receiving input from the Web-based client process. A developer inserts the application-specific statements. The resulting Web resource is stored on a Windows platform connected to the network for access by a Web server on the platform in response to a request from the Web-based client process. Using this template, robust network management

Art Unit: 2133

applications can be developed efficiently for certain platforms, such as windows platforms, with most of the application-specific development confined to generating a user interface, such as with HTML forms, to present the network information to the user and to receive user input and commands. Liu teaches (Figure 2D) in step 264, the standard network management methods are used to perform server-side verification of the data. If the data cannot pass verification, then control passes to step 266 which generates and displays an error message. If the data do pass server-side verification, then control passes to step 268 to continue processing.

Liu does not explicitly teach to use a specific error detection scheme to detect errors as stated in the present application.

However, Lubart teaches, in an analogous art, (col. 2) a recovery environment by creating the recovery environment when the operating system routines are bound together to form the kernel and at operating system initialization time. This static recovery environment exists in the operating system and provides a cross reference from the instruction address of any system failure to the appropriate recovery routine. The software or hardware error detection mechanism accesses the static cross-reference to determine *which recovery routine to initiate*. Control is then passed to the appropriate recovery routine, which has access to all registers and variables in the current context of the operating system. The system allows a nested recovery function such that each of the nested functions has an opportunity to perform any recovery action required. Each recovery routine itself may have a recovery routine. Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to utilize multiple error detection schemes within the error detection process of Liu. This modification would have been obvious to one of ordinary skill in the art because one of ordinary skill in the art would have recognized that

Art Unit: 2133

by utilizing multiple error detection schemes would allow for specific types of errors to be detected.

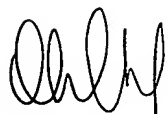
***Conclusion***

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Additional pertinent prior arts are included herein for Applicant's review.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mujtaba K. Chaudry whose telephone number is 571-272-3817. The examiner can normally be reached on Mon-Thur 9-7:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Albert DeCady can be reached on 571-272-3819. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).



Mujtaba Chaudry  
Art Unit 2133  
July 6, 2006



**GUY LAMARRE**  
**PRIMARY EXAMINER**